Optimal Function Computation in Directed and Undirected Graphs

Hemant Kowshik
CSL and Department of ECE
University of Illinois Urbana-Champaign
Email: kowshik2@illinois.edu
P. R. Kumar
CSL and Department of ECE
University of Illinois Urbana-Champaign
Email: prkumar@illinois.edu

Abstract

We consider the problem of information aggregation in sensor networks, where one is interested in computing a *function* of the sensor measurements. We allow for block processing and study in-network function computation in directed graphs and undirected graphs. We study how the structure of the function affects the encoding strategies, and the effect of interactive information exchange. Depending on the application, there could be a designated collector node, or every node might want to compute the function.

We begin by considering a directed graph $G = (\mathcal{V}, \mathcal{E})$ on the sensor nodes, where the goal is to determine the optimal encoders on each edge which achieve function computation at the collector node. Our goal is to characterize the rate region in $\mathbf{R}^{|\mathcal{E}|}$, i.e., the set of points for which there exist feasible encoders with given rates which achieve zero-error computation for asymptotically large block length. We determine the solution for directed trees, specifying the optimal encoder and decoder for each edge. For general directed acyclic graphs, we provide an outer bound on the rate region by finding the disambiguation requirements for each cut, and describe examples where this outer bound is tight.

Next, we address the scenario where nodes are connected in an undirected tree network, and every node wishes to compute a given symmetric Boolean function of the sensor data. Undirected edges permit interactive computation, and we therefore study the effect of interaction on the aggregation and communication strategies. We focus on sum-threshold functions, and determine the minimum worst-case total number of bits to be exchanged on each edge. The optimal strategy involves recursive in-network aggregation which is reminiscent of message passing. In the case of general graphs, we present a cut-set lower bound, and an achievable scheme based on aggregation along trees. For complete graphs, we prove that the complexity of this scheme is no more than twice that of the optimal scheme.

I. INTRODUCTION

Sensor networks are composed of nodes with sensing, wireless communication and computation capabilities. These networks are designed for applications like fault monitoring, data harvesting and environmental monitoring; tasks which can be broadly classified as information aggregation. In these applications, one is interested only in computing some relevant *function* of the sensor measurements. For example, one might want to compute the mean temperature for environmental monitoring, or the maximum temperature in fire alarm systems. This suggests moving away from a data-forwarding paradigm, and focusing on efficient in-network computation and communication strategies for the function of interest. This is particularly important since sensor nodes may be severely limited in terms of power and bandwidth, and can potentially generate enormous volumes of data.

There are two possible architectures for sensor networks that one might consider. First, one could designate a single collector node/fusion center which seeks to compute the function. This goal is more appropriate for data harvesting and centralized fault monitoring. Alternately, one could suppose that every node in the network wants to compute the function. The latter goal can be viewed as providing situational awareness to each sensor node, which could be very useful in applications like distributed fault monitoring, adaptive sensing and sensor-actuator networks. For example, sensor nodes might want to modify their sampling rate depending on the value of the function. We will consider both these problems.

In order to make progress on the general problem of computing functions of distributed data, we will study specific network topologies and some specific classes of functions. In this paper, we abstract out the medium access control problem associated with a wireless network, and view the network as a graph with edges representing noiseless links. The fundamental challenge is to exploit the structure of the particular function, so as to optimally combine transmissions at intermediate nodes. Thus, the problem of function computation could be regarded as being more general than finding the capacity of a wireless network. In our problem formulation, we consider the zero error block computation framework. We allow for nodes to accumulate a block of measurements and realize greater efficiency using block coding strategies. However, we

require the function to be computed with zero error for the block. To solve the problem under this framework, one needs to determine the optimal strategy for communication and computation, which includes determining the order in which nodes should transmit and the information that nodes must convey whenever they transmit. The strategy for computation may benefit from interactive information exchange between nodes, which presents an additional degree of freedom vis-a-vis the standard point-to-point communication set-up.

In Section III, we view the network as a directed graph with edges representing noiseless links. We thus consider the problem of general function computation in a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a designated collector. We focus specifically on strategies for combining information at intermediate nodes, and optimal codes for transmissions on each edge. We consider both the worst case and the average case complexity for zero error block computation with a joint probability distribution on the node measurements. Our goal is to characterize the rate region in $\mathbf{R}^{|\mathcal{E}|}$, i.e., the set of points for which there exist feasible encoders with given rates which achieve zero-error computation for large enough block length. In the case of tree graphs, we derive a necessary and sufficient condition for the encoder on each edge, which provides a complete characterization of the rate region. The extension of these results to directed acyclic graphs is more difficult. However, we provide an outer bound on the rate region by finding the disambiguation requirements for each cut, and describe examples where this outer bound is tight.

In Section IV, we address the problem of computing symmetric Boolean functions in undirected graphs. The key difference from Section III is that we consider bidirectional links and study the benefit of interaction between nodes. We show how the approach described in Section III, together with ideas from communication complexity theory, can be synthesized to develop a theory of optimal computation of symmetric Boolean functions in undirected graphs. In the case of tree networks, each edge is a cut-edge, and this allows us to derive a lower bound on the number of bits exchanged on each edge, by considering an equivalent two node problem. Further, we show that a protocol of recursive in-network aggregation along with a smart interactive coding strategy, achieves this lower bound for the class of sum-threshold functions in tree networks. The optimal strategy has a simple structure that is reminiscent of message passing, where messages

flow from the leaves towards an interior node, and then flow back from the interior node to the leaves. In the case of general graphs, we present a cut-set lower bound, and an achievable scheme based on aggregation along trees. For complete graphs, we show that the complexity of this scheme is no more than twice that of the optimal scheme.

II. RELATED WORK

In its simplest form, the problem of network function computation can be modeled as a problem of computation on graphs obtained by abstracting out the medium access control problem and channel noise. This problem is closely related to the network coding problem. Indeed, assuming independent measurements x_i and the identity function $f(x_1, x_2, \ldots, x_n) = (x_1, x_2, \ldots, x_n)$, we have the reverse of the multicast problem studied in [1]. Computing a function of independent measurements is a network *computation* problem as opposed to a network coding problem. In [2], the min-cut upper bound on the rate of computation is shown to be tight for the computation of divisible functions on tree graphs. In this paper, we generalize this result using a different approach. Further, the simplicity of the approach presented allows extensions to the case of general graphs and collocated networks.

The problem of worst-case block function computation was formulated in [3]. The authors determine the maximum rate at which a symmetric function can be computed in a random network, given the constraints of the wireless medium. They identify two classes of symmetric functions namely *type-sensitive* functions exemplified by Mean and Median, and *type-threshold* functions, exemplified by Maximum and Minimum. The maximum rates for computation of type-sensitive and type-threshold functions in random planar networks are shown to be $\Theta(\frac{1}{\log n})$ and $\Theta(\frac{1}{\log \log n})$ respectively, for a network of n nodes. A communication complexity approach was used to establish upper bounds on the rate of computation in collocated networks. Some extensions to the case of finite degree graphs are presented in [4].

In the study of the communication complexity of multi-party computation [5], one seeks to minimize the number of bits that must be exchanged in the worst case between two nodes to achieve zero-error computation of a function of the node variables. The communication

complexity of Boolean functions has been studied in [6], [7]. Further, one can consider the *direct-sum problem* [8] where several instances of the problem are considered together to obtain savings. This block computation approach is used to compute the exact complexity of the Boolean AND function in [9]. In this paper, we considerably generalize this result, which allows us to derive optimal strategies for computing more general classes of symmetric Boolean functions in undirected tree networks. The optimal communication scheme is reminiscent of message passing algorithms which have been applied very effectively to the problems of computing marginals and probabilistic inference [10], [11].

An information-theoretic formulation of this problem combines the complexity of source coding of correlated sources with rate distortion, together with the complications introduced by the function structure; see [3]. There is little or no work that addresses this most general framework. The problem of source coding with side information has been studied for the vanishing error case in [12]. This has been extended in [13] to the case where the receiver desires to know a certain function $f(X,Y)$ of the single source X and the side information Y; the authors determined the required capacity of the channel between the source and receiver to be the conditional graph entropy. However, the extension to larger networks has proved difficult. In Zero-error Information Theory, the problem of source coding with side information ensuring zero error for finite block length has been studied in [14] and [15]. The problem reduces to the task of coloring a probabilistic graph defined on the set of source samples. The minimum entropy of such a coloring approaches the graph entropy or Korner entropy, as the block length approaches infinity. Recently, the rate region for multi-round interactive function computation has been characterized for two nodes [16], and for collocated networks [17].

In this paper we do not address the problem of function computation in noisy networks. In [18], the problem of computing parity in a collocated network in the presence of noise is considered. It is shown that $O(n \log \log n)$ bits suffice to achieve correct computation with high probability. This has been extended to random planar networks in [19], where the same $\log \log n$ factor of redundancy is shown to be sufficient. Remarkably, this factor was recently shown to be tight in [20].

III. Function Computation in Directed Graphs

In this section, we abstract out the medium access control problem associated with a wireless network, and view the network as a directed graph with edges representing essentially noiseless wired links between nodes. We formulate the problem of zero error function computation on graphs. We suppose that there is a joint probability distribution on the node measurements, and allow nodes to realize greater efficiency by using block codes. We will consider both the worst case and the average case complexity for zero error block computation. Given a graph, the problem we address is to determine the set of rates on the edges which will allow zero error function computation for a large enough block length. In essence, we are exploring the interaction between the function structure and the structure of the graph; how information needs to be routed and combined at intermediate nodes to achieve certain rate vectors.

In Section III-A, we begin with the two node problem. We compute the number of bits that node v_X needs to communicate to node v_Y so that the latter can compute a function $f(X, Y)$ with zero error. For correct function computation, an encoder must disambiguate certain pairs of source symbols of node v_X, on which the function disagrees. We show by explicit construction of a code that this necessary condition is in fact sufficient. This yields the optimal alphabet and we calculate the minimum worst case and average case complexity, with the latter obtained by Huffman coding over the optimal alphabet. In Section III-B, we extend this result to directed trees with the collector as root, exploiting the fact that each edge is a cut-edge. This yields the optimal alphabet for each edge, and we separately optimize the encoders for the worst case and the average case. Thus the rate region consists of all rate points dominating a single point that is coordinate-wise optimal.

In Section III-C, we consider directed acyclic graphs. A key difference from the tree case is the presence of multiple paths to route the data, which present different opportunities to combine information at intermediate nodes. We arrive at an outer bound to the rate region by finding the disambiguation requirements for each cut of the directed graph. This outer bound is not always tight, as we show in Example 3. However, for the worst case computation of finite field parity, and the maximum or minimum functions, the outer bound is shown to be indeed tight. Further,

the only extreme points of the rate region are rate points corresponding to activating only a tree subset of edges.

A. Two Node Setting

1) Worst case complexity: We begin by considering the simple two node problem. Suppose nodes v_X and v_Y have measurements $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, where the alphabets $\mathcal{X}$ and $\mathcal{Y}$ are finite sets. Node v_X needs to optimally communicate its information to node v_Y so that a function $f(x,y)$, which takes values in $\mathcal{D}$, can be computed at v_Y with zero error. We do not consider the case where v_X and v_Y interactively compute the function. Thus node v_X has an encoder $\mathcal{C} : \mathcal{X} \rightarrow \{0,1\}^*$, which maps its measurement x to the codeword $\mathcal{C}(x)$, and node v_Y has a decoder $g : \{0,1\}^* \times \mathcal{Y} \rightarrow \mathcal{D}$ which maps the received codeword $\mathcal{C}(x)$ and its own measurement y to a function estimate, $g(\mathcal{C}(x), y)$. The set of all possible codewords is called the codebook, denoted by $\mathcal{C}(\mathcal{X})$

Definition 1 (Feasible Encoder): An encoder $\mathcal{C}$ is *feasible* if there exists a decoder $g : \{0,1\}^* \times \mathcal{Y} \rightarrow \mathcal{D}$ such that $g(\mathcal{C}(x), y) = f(x,y)$ for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$. Thus, a feasible encoder is one that achieves error-free function computation.

Theorem 1 (Characterization of Feasible Encoders): An encoder $\mathcal{C}$ is feasible if and only if given $x^1, x^2 \in \mathcal{X}$, $\mathcal{C}(x^1) = \mathcal{C}(x^2)$ implies $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$.

Proof: By definition, if $\mathcal{C}$ is a feasible encoder, then there exists a corresponding decoder g such that $g(\mathcal{C}(x^1), y) = f(x^1, y)$ and $g(\mathcal{C}(x^2), y) = f(x^2, y)$, for all $y \in \mathcal{Y}$. Further, if $\mathcal{C}(x^1) = \mathcal{C}(x^2)$, we have $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$.

To prove the converse, we need to construct a decoding function $g : \{0,1\}^* \times \mathcal{Y} \rightarrow \mathcal{D}$. For each codeword C^* in the codebook, define $\mathcal{C}^{-1}(C^*) := \{x \in \mathcal{X} : \mathcal{C}(x) = C^*\}$. For fixed $y \in \mathcal{Y}$ and fixed codeword $C^* \in \mathcal{C}(\mathcal{X})$, the decoder mapping is given by $g(C^*, y) := f(x^{nom}(C^*), y)$ for any arbitrary $x^{nom}(C^*) \in \mathcal{C}^{-1}(C^*)$. We show that this decoder works for any fixed x and y. Indeed, $g(\mathcal{C}(x), y) = f(x^{nom}, y)$ where $x^{nom} \in \mathcal{C}^{-1}(\mathcal{C}(x))$. Thus, $\mathcal{C}(x^{nom}) = \mathcal{C}(x)$ and by assumption $f(x^{nom}, y) = f(x,y)$. Hence, $g(\mathcal{C}(x), y) = f(x^{nom}, y) = f(x,y)$ for all $y \in \mathcal{Y}$. $\square$

Any feasible encoder $\mathscr{C}$ can be viewed as partitioning the set $\mathscr{X}$ into $\Pi(\mathscr{C}) := \{S_1, S_2, \ldots, S_k\}$ such that for $x^1 \in C_i, x^2 \in C_j$, we have $\mathscr{C}(x^1) = \mathscr{C}(x^2)$ if and only if $i = j$. Define an equivalence relation "$\leftrightarrow$" between $x^1, x^2 \in \mathscr{X}$ by:

$$x^1 \leftrightarrow x^2 \text{ if and only if } f(x^1, y) = f(x^2, y) \text{ for all } y \in \mathscr{Y}.$$

Consider the encoder $\mathscr{C}^{OPT}$ which assigns a distinct codeword to each resulting equivalence class. Clearly, $\mathscr{C}^{OPT}$ is a feasible encoder, since $\mathscr{C}^{OPT}(x^1) = \mathscr{C}^{OPT}(x^2)$ implies $x^1 \leftrightarrow x^2$, and hence $f(x^1, y) = f(x^2, y)$ for all $y \in \mathscr{Y}$. $\mathscr{C}^{OPT}$ is optimal in the sense that any other feasible encoder $\mathscr{C}$ must have at least as many codewords as $\mathscr{C}^{OPT}$:

Theorem 2 (Optimality of $\mathscr{C}^{OPT}$): Let $\Pi(\mathscr{C}^{OPT}) := \{S_1^{OPT}, S_2^{OPT}, \ldots, S_k^{OPT}\}$ be the partition of $\mathscr{X}$ generated by $\mathscr{C}^{OPT}$, and let $\Pi(\mathscr{C}) := \{S_1, S_2, \ldots, S_l\}$ be the partition of $\mathscr{X}$ generated by any other feasible encoder $\mathscr{C}$. Then,

(i) $\Pi(\mathscr{C})$ must be a finer partition than $\Pi(\mathscr{C}^{OPT})$.

(ii) The minimum number of bits that node v_X needs to communicate is $\lceil \log |\Pi(\mathscr{C}^{OPT})| \rceil$.

Proof: First we claim that any subset $S_i \in \Pi(\mathscr{C})$ can have nonempty intersection with exactly one subset $S_j^{OPT} \in \Pi(\mathscr{C}^{OPT})$. Suppose not. Then there exist $x^1, x^2 \in S_i$ such that $x^1 \in S_{j_1}^{OPT}$ and $x^2 \in S_{j_2}^{OPT}$. Since $\mathscr{C}(x^1) = \mathscr{C}(x^2)$, by Theorem 1, we must have $f(x^1, y) = f(x^2, y)$ for all $y \in \mathscr{Y}$. However, by construction of $\mathscr{C}^{OPT}$, x^1 and x^2 must belong to distinct equivalence classes i.e., $x^1 \not\leftrightarrow x^2$. Hence, there exists y^* such that $f(x^1, y^*) \neq f(x^2, y^*)$, which is a contradiction. This shows that the partition generated by any encoder $\mathscr{C}$ must be a further subdivision of the partition generated by $\mathscr{C}^{OPT}$, i.e., finer than $\Pi(\mathscr{C}^{OPT})$. So node v_X needs to communicate at least $\lceil \log |\Pi(\mathscr{C}^{OPT})| \rceil$ bits. $\square$

We can extend this to the case where v_X collects a block of N measurements $\underline{x} = (x_1, x_2, \ldots, x_N) \in \mathscr{X}^N$, and v_Y collects a block of N measurements $\underline{y} = (y_1, y_2, \ldots, y_N) \in \mathscr{Y}^N$. We want to find a block encoder $\mathscr{C}^N : \mathscr{X}^N \to \{0, 1\}^*$ so that the vector function $f^{(N)}(\underline{x}, \underline{y}) = (f(x_1, y_1), \ldots, f(x_N, y_N))$ can be computed without error, for all $\underline{x} \in \mathscr{X}^N, \underline{y} \in \mathscr{Y}^N$. All the above results carry over to the error-free block computation case. As before, we define an equivalence $\leftrightarrow$ between $\underline{x}^1, \underline{x}^2 \in \mathscr{X}^N$ if $f^{(N)}(\underline{x}^1, \underline{y}) = f^{(N)}(\underline{x}^2, \underline{y})$ for all $\underline{y} \in \mathscr{Y}^N$. The optimal encoder $\mathscr{C}^{N,OPT}$ is once again obtained

by assigning distinct codewords to each equivalence class. Since we are stringing together N independent instances, we have $|\Pi(\mathscr{C}^{N,OPT})| = |\Pi(\mathscr{C}^{OPT})|^N$. Hence the minimum number of bits per computation that node v_X needs to communicate is $\frac{\lceil N\log|\Pi(\mathscr{C}^{OPT})|\rceil}{N}$ which converges to $\log|\Pi(\mathscr{C}^{OPT})|$ as $N \to \infty$.

2) Average case complexity: Suppose now that the measurements X, Y are drawn from the joint probability distribution $p(X,Y)$, with the goal being to minimize the average number of bits that need to be communicated, i.e., the *average case complexity*.

Definition 2 (Feasible Encoder): An encoder $\mathscr{C} : \mathscr{X} \to \{0,1\}^*$ is *feasible* if there exists a decoder $g : \{0,1\}^* \times \mathscr{Y} \to \mathscr{D}$ such that $g(\mathscr{C}(x),y) = f(x,y)$ for all $\{(x,y) \in \mathscr{X} \times \mathscr{Y} : p(x,y) > 0\}$.

Theorem 3: An encoder $\mathscr{C}$ is feasible if and only if, given $x^1, x^2 \in \mathscr{X}$, $\mathscr{C}(x^1) = \mathscr{C}(x_2)$ implies $f(x_1,y) = f(x_2,y)$ for $\{y \in \mathscr{Y} : p(x_1,y)p(x_2,y) > 0\}$.

Proof: By definition, if $\mathscr{C}$ is a feasible encoder, then there exists a corresponding decoder g such that $g(\mathscr{C}(x^1),y) = f(x^1,y)$ and $g(\mathscr{C}(x^2),y) = f(x^2,y)$, for all $\{y \in \mathscr{Y} : p(x^1,y)p(x^2,y) > 0\}$. Further, if $\mathscr{C}(x^1) = \mathscr{C}(x^2)$, we have $f(x^1,y) = f(x^2,y)$ for $\{y \in \mathscr{Y} : p(x^1,y)p(x^2,y) > 0\}$.

To prove the converse, we need to construct a decoding function $g : \{0,1\}^* \times \mathscr{Y} \to \mathscr{D}$. For each codeword $\mathscr{C}^*$ in the codebook, define $\mathscr{C}^{-1}(C^*) := \{x \in \mathscr{X} : \mathscr{C}(x) = C^*\}$. For fixed $y \in \mathscr{Y}$ and fixed codeword $\mathscr{C}^* \in \mathscr{C}(\mathscr{X})$, the decoder mapping is given by $g(C^*,y) := f(x^{nom}(C^*,y),y)$ for any arbitrary $x^{nom}(C^*,y) \in \mathscr{C}^{-1}(C^*)$ with $p(x^{nom}(C^*,y),y) > 0$. We show that this decoder works for any fixed x and y with $p(x,y) > 0$. Indeed, $g(\mathscr{C}(x),y) = f(x^{nom},y)$ where $x^{nom} \in \mathscr{C}^{-1}(\mathscr{C}(x))$ with $p(x^{nom},y) > 0$. Thus, $\mathscr{C}(x^{nom}) = \mathscr{C}(x)$ and by assumption $f(x^{nom},y) = f(x,y)$ since $p(x^{nom},y)p(x,y) > 0$. Hence, $g(\mathscr{C}(x),y) = f(x^{nom},y) = f(x,y)$. $\square$

We now define "$x^1 \leftrightarrow x^2$" when $f(x^1,y) = f(x^2,y)$ for $\{y \in \mathscr{Y} : p(x^1,y)p(x^2,y) > 0\}$. Now the $\leftrightarrow$ relation is reflexive and symmetric, but not necessarily transitive. However, if $p(x,y) > 0$ for all $(x,y) \in \mathscr{X} \times \mathscr{Y}$, then $\leftrightarrow$ is an equivalence relation. We can construct an encoder $\mathscr{C}^{OPT}$ which assigns a distinct codeword to each equivalence class. Let $\Pi(\mathscr{C}^{OPT}) := \{S_1^{OPT}, S_2^{OPT}, \ldots, S_k^{OPT}\}$ be the partition of $\mathscr{X}$ generated by $\mathscr{C}^{OPT}$. Analogous to Theorem 2, we can show that the encoder $\mathscr{C}^{OPT}$ has the *optimal alphabet* $\mathscr{A}$, with the probability distribution vector $\underline{q} = \{q_1, q_2, \ldots, q_k\}$ where $q_i := \sum_{x \in S_i^{OPT}} \sum_{y \in \mathscr{Y}} p(x,y)$.

Once the optimal alphabet is fixed, the optimal code $\mathscr{C}^{OPT}$ is the *binary Huffman code* for the probability vector $\underline{q}$. Since the Huffman code has an average code length within one bit of the entropy,

$$H(q_1, q_2, \ldots, q_k) \leq E[l(\mathscr{C}^{OPT})] \leq H(q_1, q_2, \ldots, q_k) + 1.$$

The extension to the case where nodes v_X, v_Y collect a block of N i.i.d. measurements is straightforward. The optimal alphabet is $\mathscr{A}^N$, which has the product distribution $\underline{q}^N$. The optimal encoder is obtained via the Huffman code for the optimal alphabet. Its expected length satisfies

$$\frac{H(\underline{q}^N)}{N} \leq \frac{E[l(\mathscr{C}^{N,OPT})]}{N} \leq \frac{H(\underline{q}^N) + 1}{N}.$$

Hence the minimum number of bits per computation that node v_X needs to communicate converges to $H(\underline{q})$ as $N \to \infty$.

B. Function Computation in Directed Trees

Let us now consider computation on a *tree graph*. Consider a directed tree $G = (\mathscr{V}, \mathscr{E})$ with nodes $\mathscr{V} := \{v_1, v_2, \ldots, v_n\}$ and root node v_1. Edges represent communication links, so that node v_j can transmit to node v_i if $(v_j, v_i) \in \mathscr{E}$. Each node v_i makes a measurement $x_i \in \mathscr{X}_i$, and the collector node v_1 wants to compute a function $f(x_1, x_2, \ldots, x_n)$ with no error. We seek to minimize the worst case complexity on each edge.

For each node i, let $\pi(v_i)$ be the unique node to which node i has an outgoing edge, and let $\mathscr{N}^-(v_i) := \{v_j \in \mathscr{V} : (v_j, v_i) \in \mathscr{E}\}$. The *height* of a node v_i is the length of the longest directed path from a leaf node to v_i. Define the *descendant set* $D(v_i)$ to be the subset of nodes in $\mathscr{V}$ from which there exist directed paths to node v_i. The graph induced on $D(v_i)$ is a tree with node v_i as root. Each node transmits exactly once and the computation proceeds in a bottom-up fashion, starting from the leaf nodes and proceeding up the tree.

Each leaf node v_i has an encoder $\mathscr{C}_i : \mathscr{X}_i \to \{0,1\}^*$ that maps its measurement x_i to a codeword $\mathscr{C}_i(x_i)$ which is transmitted on the edge $(v_i, \pi(v_i))$. Each non-leaf node v_j for $j \neq 1$ has an encoder $\mathscr{C}_j$ which maps its measurement x_j as well as the codewords received from $\mathscr{N}^-(v_j)$, to

a codeword transmitted on the edge $(v_j, \pi(v_j))$. Thus the computation proceeds in a bottom-up fashion. Let C_i denote the codeword transmitted by node v_i, and $C_S := \{C_i : v_i \in S\}$ denote the set of codewords transmitted by nodes in S.

Definition 3: A set of encoders $\{\mathscr{C}_i : 2 \leq i \leq n\}$ is said to be feasible if there is a decoding function g_1 at the collector node v_1 such that $g(x_1, \mathscr{C}_{\mathscr{N}^-(v_1)}) = f(x_1, x_2, \ldots, x_n)$ for all $(x_1, x_2, \ldots, x_n) \in \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n$.

Lemma 1: If a set of encoders $\{\mathscr{C}_i : 2 \leq i \leq n\}$ is feasible, then the encoder $\mathscr{C}_i$ at node v_i must separate[1] $x^1_{D(v_i)} \in \mathscr{X}_{D(v_i)}$ from $x^2_{D(v_i)} \in \mathscr{X}_{D(v_i)}$, if there exists an assignment $x^*_{\mathscr{V} \backslash D(v_i)}$ such that $f(x^1_{D(v_i)}, x^*_{\mathscr{V} \backslash D(v_i)}) \neq f(x^2_{D(v_i)}, x^*_{\mathscr{V} \backslash D(v_i)})$.

Proof: The removal of edge $(v_i, \pi(v_i))$ separates the graph into two disconnected subtrees $D(v_i)$ and $\mathscr{V} \backslash D(v_i)$. We combine all the nodes in $D(v_i)$ into a supernode v_α, and all the nodes in $\mathscr{V} \backslash D(v_i)$ into a supernode v_β. The result now follows from Theorem 1. $\square$

To prove the converse, we explicitly define the encoders $\mathscr{C}_2, \mathscr{C}_3, \ldots, \mathscr{C}_n$ and a decoding function g, and prove that it achieves correct function computation. Define the alphabet for encoder $\mathscr{C}_i$ on edge $(v_i, \pi(v_i))$ as,

$$\mathscr{A}_i := \{h_i : \mathscr{X}_{\mathscr{V} \backslash D(v_i)} \to \mathscr{D} \text{ s. t. } \exists x^*_{D(v_i)} \in \mathscr{X}_{D(v_i)},$$

$$h_i(x_{\mathscr{V} \backslash D(v_i)}) = f(x^*_{D(v_i)}, x_{\mathscr{V} \backslash D(v_i)}) \quad \forall x_{\mathscr{V} \backslash D(v_i)} \in \mathscr{X}_{\mathscr{V} \backslash D(v_i)}\}.$$

Thus codewords sent by node v_i can be viewed as *normal forms* on variables $X_{\mathscr{V} \backslash D(v_i)}$, or as partial functions on $X_{\mathscr{V} \backslash D(v_i)}$.

Encoder at node v_i**:** On receiving the codeword corresponding to $h_j : \mathscr{X}_{\mathscr{V} \backslash D(v_j)} \to \mathscr{D}$, on incoming edge (v_j, v_i), node v_i assigns nominal values, $x^{nom}_{D(v_j)}$ to variables $X_{D(v_j)}$ such that

$$f(x^{nom}_{D(v_j)}, x_{\mathscr{V} \backslash D(v_j)}) = h_j(x_{\mathscr{V} \backslash D(v_j)}) \quad \forall x_{\mathscr{V} \backslash D(v_j)} \in \mathscr{X}_{\mathscr{V} \backslash D(v_j)}. \tag{1}$$

Given nominal values for all nodes in $D(v_i) \backslash \{v_i\}$, and its own measurement x_i, node v_i substitutes

---

[1]Node v_i does not have access to $x_{D(v_i)}$ directly but only the codewords received from $\mathscr{N}^-(v_i)$. When we say that the encoder $\mathscr{C}_i$ must separate $x_{D(v_i)}, \tilde{x}_{D(v_i)}$, we are considering $\mathscr{C}_i$ as an implicit function of $x_{D(v_i)}$.

these values to obtain a function $h_i : \mathcal{X}_{\mathcal{V} \setminus D(v_i)} \to \mathcal{D}$ such that

$$h_i(x_{\mathcal{V} \setminus D(v_i)}) = f(x^{nom}_{D(v_i) \setminus \{v_i\}}, x_i, x_{\mathcal{V} \setminus D(v_i)}) \text{ for all } x_{\mathcal{V} \setminus D(v_i)} \in \mathcal{X}_{\mathcal{V} \setminus D(v_i)}.$$

If $v_i \neq v_1$, node v_i then transmits the codeword $\mathcal{C}_i$ corresponding to function $h_i \in \mathcal{A}_i$ on the edge $(v_i, \pi(v_i))$.

Decoding function g: The collector node v_1 assigns nominal values to the variables $X_{D(v_1) \setminus \{v_1\}}$. The decoding function g is given by $g(x_1, C_{\mathcal{N}^-(v_1)}) := h_1 = f(x_1, x^{nom}_{D(v_1) \setminus \{v_1\}})$.

Theorem 4: Let $x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix}$ be any fixed assignment of node values. Let the encoders at node $v_2, v_3, \ldots, v_n$ be as above. Then function h_i computed by node v_i is,

$$h_i(x_{\mathcal{V} \setminus D(v_i)}) = f(x^{fix}_{D(v(i))}, x_{\mathcal{V} \setminus D(v_i)}) \qquad \forall x_{\mathcal{V} \setminus D(v_i)} \in \mathcal{X}_{\mathcal{V} \setminus D(v_i)}.$$

Consequently the decoding function g satisfies $g(x_1^{fix}, \mathcal{C}_{\mathcal{N}^-(v_1)}) = f(x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix})$.

Proof: The proof proceeds by induction. The theorem is trivially true for all leaf nodes v_i, since by assumption $h_i(x_{\mathcal{V} \setminus D(v_i)}) = f(x_{v_i}^{fix}, x_{\mathcal{V} \setminus D(v_i)})$ for all $x_{\mathcal{V} \setminus D(v_i)} \in \mathcal{X}_{\mathcal{V} \setminus D(v_i)}$. Suppose it is true for all nodes with height less than κ. Consider a node v_i with height κ. All the nodes in $\mathcal{N}^-(v_i)$ must have height less than κ. On receiving the codeword corresponding to h_j on edge (v_j, v_i), node v_i assigns nominal values to variables in $X_{D(v_j)}$ so that (1) is satisfied. From the induction assumption, we have

$$h_j(x_{\mathcal{V} \setminus D(v_j)}) = f(x^{fix}_{D(v_j)}, x_{\mathcal{V} \setminus D(v_j)}) \quad \forall x_{\mathcal{V} \setminus D(v_j)} \in \mathcal{X}_{\mathcal{V} \setminus D(v_j)}. \tag{2}$$

Since (2) is true for all $v_j \in \mathcal{N}^-(v_i)$, we can simultaneously substitute the nominal values $x^{nom}_{D(v_i) \setminus \{v_i\}}$ for the variables $X_{D(v_i) \setminus \{v_i\}}$ and the value x_i^{fix} for the variable $X_{\{v_i\}}$, to obtain a function h_i satisfying

$$\begin{aligned} h_i(x_{\mathcal{V} \setminus D(v_i)}) &= f(x^{nom}_{D(v(i)) \setminus \{v_i\}}, x^{fix}_{v_i}, x_{\mathcal{V} \setminus D(v_i)}) \quad \forall x_{\mathcal{V} \setminus D(v_i)} \\ &= f(x^{fix}_{D(v(i))}, x_{\mathcal{V} \setminus D(v_i)}) \qquad \forall x_{\mathcal{V} \setminus D(v_i)}, \end{aligned} \tag{3}$$

where (3) follows from (1) and (2). This establishes the induction step and completes the proof.

For the special case of the collector node v_i, we have

$$g(x_1^{fix}, \mathscr{C}_{\mathscr{N}^-(v_1)}) = h_1 = f(x_{D(v_1)}^{fix}) = f(x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix}).$$

Since this is true for every fixed assignment of the node values, we can achieve error-free computation of the function. Hence the set of encoders described above is feasible. $\square$

For node v_i, consider the equivalence relation "$\leftrightarrow_i$" where $x_{D(v_i)}^1 \leftrightarrow_i x_{D(v_i)}^2$ if $f(x_{D(v_i)}^1, x_{\mathscr{V} \setminus D(v_i)}) = f(x_{D(v_i)}^2, x_{\mathscr{V} \setminus D(v_i)})$ for all $x_{\mathscr{V} \setminus D(v_i)} \in \mathscr{X}_{\mathscr{V} \setminus D(v_i)}$. It is easy to check that the equivalence classes generated by $\leftrightarrow_i$ are captured exactly by the alphabet $\mathscr{A}_i$. Thus the above encoders use exactly the optimal alphabet. Hence, the minimum worst case complexity for encoder $\mathscr{C}_i$ is $\lceil \log(|\mathscr{A}_i|) \rceil$ on the edge $(v_i, \pi(v_i))$.

The extension to the case where node v_i collects a block of N independent measurements $\underline{X}_i \in \mathscr{X}_i^N$, and the collector node v_1 wants to compute the vector function $f^{(N)}(\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n)$, is straightforward. We can thus achieve a minimum worst case complexity arbitrarily close to $\log|\mathscr{A}_i|$ bits for encoder $\mathscr{C}_i$. It should be noted that the minimum worst case complexity of encoder $\mathscr{C}_i$ does not depend on the encoders of the other nodes.

If there is a probability distribution $p(X_1, X_2, \ldots, X_n)$ on the measurements, then we can obtain a necessary and sufficient condition by considering all edge cuts.

Lemma 2: Consider a cut which partitions the nodes into S and $\mathscr{V} \setminus S$ with $v_1 \in \mathscr{V} \setminus S$. Let $\delta^+(S)$ be the set of all edges from nodes in S to nodes in $\mathscr{V} \setminus S$. Then the set of encoders $\{\mathscr{C}_i : 2 \leq i \leq n\}$ is feasible if and only if for every cut, the encoder on at least one of the edges in $\delta^+(S)$ separates $x_S^1, x_S^2 \in \mathscr{X}_S$ if there exists an assignment $x_{\mathscr{V} \setminus S}^*$ such that $f(x_S^1, x_{\mathscr{V} \setminus S}^*) \neq f(x_S^2, x_{\mathscr{V} \setminus S}^*)$ and $p(x_S^1, x_{\mathscr{V} \setminus S}^*) p(x_S^2, x_{\mathscr{V} \setminus S}^*) > 0$.

Proof: Necessity is as before. For the converse, suppose the set of encoders is not feasible. Then there exist assignments $(x_1^*, x_{\mathscr{V} \setminus v_1}^A)$ and $(x_1^*, x_{\mathscr{V} \setminus v_1}^B)$ such that $f(x_1^*, x_{\mathscr{V} \setminus v_1}^A) \neq f(x_1^*, x_{\mathscr{V} \setminus v_1}^B)$ and $p(x_1^*, x_{\mathscr{V} \setminus v_1}^A) p(x_1^*, x_{\mathscr{V} \setminus v_1}^B) > 0$. However, the codewords received from nodes in $\mathscr{N}^-(v_1)$ are the same for both assignments. For the cut which separates v_1 from $\mathscr{V} \setminus v_1$, there is no encoder on $\delta^+(S)$ which separates $x_{\mathscr{V} \setminus v_1}^A$ and $x_{\mathscr{V} \setminus v_1}^B$. $\square$

The above proof of the converse is not constructive. The construction is much harder now

since the encoders are coupled, as shown by the following example.

Example 1: Consider the three node network $G = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, v_2, v_3\}$ and $\mathcal{E} = \{(v_2, v_1), (v_3, v_1)\}$ (see Figure 1(a)). Let $\mathcal{X}_1 = \{x^{1a}\}$, $\mathcal{X}_2 = \{x^{2a}, x^{2b}\}$, $\mathcal{X}_3 = \{x^{3a}, x^{3b}\}$. Suppose $p(x^{1a}, x^{2a}, x^{3a}) = p(x^{1a}, x^{2b}, x^{3b}) = \frac{1}{2}$. The function is given by $f(X_1, X_2, X_3) = (X_1, X_2, X_3)$. Considering the cut $(\{v_2, v_3\}, \{v_1\})$, either v_2 or v_3 needs to separate its two values. Thus the two encoders are no longer independent.
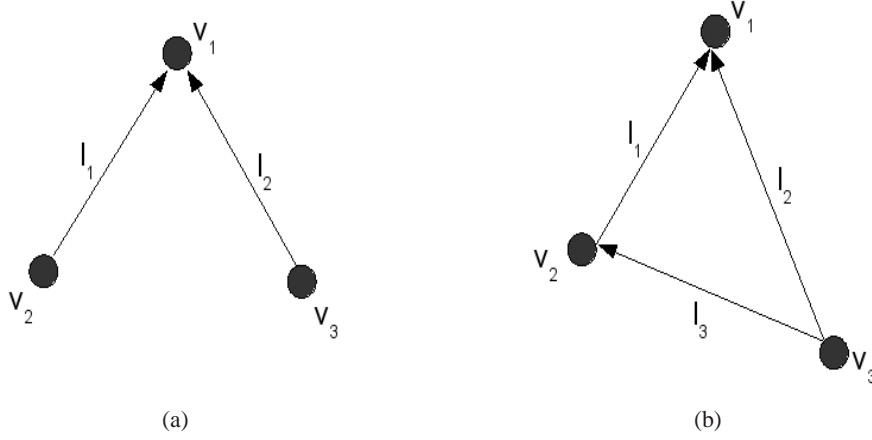

(a)   (b)

Fig. 1.   Two simple networks of Examples 1 and 2

In general, we can trade off between the encoders on different edges. However, if we assume that $p(x_1, x_2, \ldots, x_n) > 0$ for all $(x_1, x_2, \ldots, x_n)$, we can separately minimize the average description length of each encoder. The optimal encoder constructs a Huffman code on the optimal alphabet $\mathcal{A}_i$. Suppose $\underline{q}_i$ is the probability vector induced on the alphabet $\mathcal{A}_i$. Then, by taking long blocks of measurements, we can achieve a minimum average case complexity arbitrarily close to $H(\underline{q}_i)$ for encoder $\mathcal{C}_i$.

C. Function Computation in Directed Acyclic Graphs

The extension from trees to directed acyclic graphs presents significant challenges, since there is no longer a unique path from every node to the collector. Consider a weakly connected directed acyclic graph (DAG) $G = (\mathcal{V}, \mathcal{E})$, where each node v_i collects a block of N measurements

$\underline{X}_i \in \mathcal{X}_i^N$. The collector node v_1 is the unique node with only incoming edges, which wants to compute the vector function $f^N(\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n)$ with zero error.

Let the encoder mapping on edge (v_j, v_i) be denoted by $\mathscr{C}_{ji}^N$, which maps the measurement vector $\underline{X}_j$ and the codewords received thus far, to a codeword transmitted on edge (v_j, v_i). Since there are no cycles in G, function computation proceeds in a bottom-up fashion. Node v_i receives codewords $\mathscr{C}_{ji}^N$ on each incoming edge (v_j, v_i) and then transmits a codeword $\mathscr{C}_{ik}$ on each outgoing edge (v_i, v_k). A set of encoders is said to be feasible if there is a decoding function at the collector node v_1 which maps the received codewords to the correct function value. Let $l_{wc}(\mathscr{C}_{ij}^N)$ and $l_{avg}(\mathscr{C}_{ij}^N)$ denote the worst case and average case complexity, respectively, of the encoder $\mathscr{C}_{ij}^N$. The rate of encoder $\mathscr{C}_{ij}^N$ is

$$R_{wc}(\mathscr{C}_{ij}^N) = \frac{l_{wc}(\mathscr{C}_{ij}^N)}{N} \text{ and } R_{avg}(\mathscr{C}_{ij}^N) = \frac{l_{avg}(\mathscr{C}_{ij}^N)}{N}.$$

Thus we can assign a rate vector in $\mathbf{R}^{|\mathscr{E}|}$ to every feasible set of encoders. Let $\mathscr{R}_{wc}^{(N)}$ in the worst case (or $\mathscr{R}_{avg}^{(N)}$ in the average case) be the set of feasible rate vectors for encoders of block length N. Then the rate region $\mathscr{R}_{wc}$ (or $\mathscr{R}_{avg}$) is given by the closure in $\mathbf{R}^{|\mathscr{E}|}$ of the finite block length rate vectors:

$$\mathscr{R}_{wc} := \overline{\bigcup_{N \geq 1} \mathscr{R}_{wc}^{(N)}} \text{ and } \mathscr{R}_{avg} := \overline{\bigcup_{N \geq 1} \mathscr{R}_{avg}^{(N)}}.$$

Consider the following example.

Example 2: We have three nodes $\{v_1, v_2, v_3\}$ connected as shown in Figure 1(b). Let $\mathcal{X}_1 = \mathcal{X}_2 = \mathcal{X}_3 = \{0, 1, 2, 3\}$, and suppose node v_1 wants to compute $f(X_1, X_2, X_3) = (X_1 + X_2 + X_3) mod 4$. It is easy to check that $(2, 0, 2)$ and $(2, 2, 0)$ are feasible rate vectors for (l_1, l_2, l_3). These are rate vectors associated with the two tree subgraphs. Further, one can also check that $(2, 1, 1)$ is

1) Outer bound on the rate region: Consider any cut of the graph G which partitions nodes into subsets S and $\mathcal{V} \setminus S$ with $v_1 \in \mathcal{V} \setminus S$. Let $\delta^+(S)$ be the set of edges from some node in S to some node in $\mathcal{V} \setminus S$.

Lemma 3: Consider a set of encoders which achieve error free block function computation

with rate vector $\{R_{wc}(i,j)\}_{(v_i,v_j)\in\mathscr{E}}$. Given any assignments $\underline{x}_S^1$ and $\underline{x}_S^2$ of the nodes in S, if there exists an assignment $\underline{x}_{\mathscr{V}\setminus S}$ such that $f^{(N)}(\underline{x}_{\mathscr{V}\setminus S},\underline{x}_S^1)\neq f^{(N)}(\underline{x}_{\mathscr{V}\setminus S},\underline{x}_S^2)$, then the encoders on at least one of the edges in $\delta^+(S)$ must separate $\underline{x}_S^1$ and $\underline{x}_S^2$.

(i) In the worst case block computation scenario, an outer bound on the rate region is given by

$$\sum_{(v_i,v_j)\in\delta^+(S)} R_{ij} \geq \log|\Pi(\mathscr{C}_S^1)| \text{ for all cuts } (S,\mathscr{V}\setminus S),$$

where $\Pi(\mathscr{C}_S^1)$ is the partition of $\mathscr{X}_S$ into the appropriate equivalence classes.

(ii) Suppose we have a probability distribution with $p(X_1,X_2,\ldots,X_n)>0$. Given a cut $(S,\mathscr{V}\setminus S)$, let $R\subset\mathscr{V}\setminus S$ be the subset of nodes which have a directed path to some node in S. In the average case block computation scenario, an outer bound on the rate region is given by

$$\sum_{(v_i,v_j)\in\delta^+(S)} R_{ij} \geq H([X_S]|X_R) \text{ for all cuts } (S,\mathscr{V}\setminus S),$$

where $[X_S]|X_R$ is the equivalence class to which X_S belongs, given X_R and a particular function.

2) Achievable region:

Lemma 4: Consider any directed tree subgraph G_T with root node v_1. Let us suppose that only the edges in G_T can be used for communication. Then we can construct encoders on each edge, which minimize worst case or average case complexity. The rate vector corresponding to a tree G_T is the limit of the rate vectors for the optimal finite block length encoders for G_T. Thus, for a given tree G_T:

(i) The worst case rate vector corresponding to the tree G_T is an extreme point of the worst case rate region $\mathscr{R}_{wc}$.

(ii) If $p(x_1,x_2,\ldots,x_n)>0$ for all $(x_1,x_2,\ldots,x_n)$, the rate vector corresponding to the tree G_T is an extreme point of the average case rate region $\mathscr{R}_{avg}$.

The convex hull of the rate points corresponding to trees is achievable. However, we do not know if these are the *only* extreme points of the rate region $\mathscr{R}$.

3) Some examples:

Example 3 (Arithmetic Sum): Consider three nodes v_1, v_2, v_3 connected as in Figure 1(b). Let $\mathscr{X}_2 = \mathscr{X}_3 = \{0,1\}$, with node v_1 having no measurements. Suppose node v_1 wants to compute $f(X_1, X_2, X_3) = X_2 + X_3$. Let (R_{21}, R_{31}, R_{32}) be the rate vector associated with edges (l_1, l_2, l_3). The outer bound on $\mathscr{R}_{wc}$ is:

$$R_{21} \geq 1; \quad R_{21} + R_{31} \geq \log 3; \quad R_{32} + R_{31} \geq 1.$$

The subset of the rate region achievable by trees is:

$$R_{21} = \lambda + (1-\lambda)\log 3, R_{31} = \lambda, R_{32} = (1-\lambda) \text{ for } 0 \leq \lambda \leq 1.$$

Suppose that X_1, X_2 are i.i.d. with $p(X_1 = 0) = p(X_1 = 1) = 0.5$. The outer bound on $\mathscr{R}_{avg}$ is:

$$R_{21} \geq 1; \quad R_{21} + R_{31} \geq \frac{3}{2}; \quad R_{32} + R_{31} \geq 1.$$

The subset of the rate region achievable by trees is:

$$R_{21} = \lambda + (1-\lambda)\frac{3}{2}, R_{31} = \lambda, R_{32} = (1-\lambda) \text{ for } 0 \leq \lambda \leq 1.$$

Example 4 (Finite field parity): Let $\mathscr{X}_i = \{0, 1, \ldots, D-1\}$ for each node v_i. Suppose the collector node v_1 wants to compute the function $(X_1 + X_2 + \ldots + X_n) \bmod D$. In this case, the outer bound on the worst case rate region described in Lemma 3 is tight. Indeed, since the set of all outgoing links from a node is a valid cut, we have $\sum_{(v_i,v_j)\in\mathscr{E}} R_{ij} \geq \log_2 D$

An obvious achievable strategy is for every leaf node v_i to split its block and transmit it on the outgoing edges from v_i. Next, we move to a node at height 1. This node receives partial blocks from various leaf nodes, and can hence compute an intermediate parity for some instances of the block. It then splits its block along the various outgoing edges. The crucial point is that the worst case description length per instance remains $\log_2 D$. Proceeding recursively up the DAG, we see that we can achieve the outer bound.

Example 5 (Max/Min): Let $\mathscr{X}_i = \{0, 1, \ldots, D-1\}$ for each node v_i. Suppose the collector node v_1 wants to compute $max(X_1, X_2, \ldots, X_n)$. The outer bound to the worst case rate region described

in Lemma 3 is tight. The achievable strategy is similar to the parity case, where nodes compute intermediate maximum values and split their blocks on the outgoing edges. Once again, we utilize the fact that the range of the Max function remains constant irrespective of the number of nodes.

IV. COMPUTING SYMMETRIC BOOLEAN FUNCTIONS IN UNDIRECTED GRAPHS

In this section, we address the problem of symmetric Boolean function computation in an undirected graph. Each node has a Boolean variable and *all* nodes want to compute a given symmetric Boolean function. As in Section III, we adopt a deterministic framework and consider the problem of worst case block computation. Further, since the graph is undirected, the set of admissible strategies includes all interactive strategies, where a node may exchange several messages with other nodes, with node i's transmission being allowed to depend on all previous transmissions heard by node i, and node i's block of measurements. This is in contrast with the problem studied in Section III.

We begin by reviewing a toy problem from [9] where the exact communication complexity of the AND function of two variables is shown to be $\log_2 3$ bits, for block computation. In Section IV-A, we generalize this approach to the two node problem, where each node i has an integer variable X_i and both nodes want to compute a function $f(X_1, X_2)$ which only depends on $X_1 + X_2$. We derive an optimal single-round strategy for the class of sum-threshold functions, which evaluate to 1 if $X_1 + X_2$ exceeds a threshold, and an approximate strategy for the class of sum-interval functions, which evaluate to 1 if $a \leq X_1 + X_2 \leq b$, the upper and lower bounds do not match. The general achievable strategy involves separation of the source alphabet, followed by coding, and can be used for any general function.

In Section IV-B, we consider symmetric Boolean function computation on trees. Since every edge is a cut-edge, we can obtain a cut-set lower bound for the number of bits that must be exchanged on an edge, by reducing it to a two node problem with general alphabets. For the class of sum-threshold functions, we are able to match the cut-set bound by constructing an achievable strategy that is reminiscent of message passing algorithms. In Section IV-D, for general graphs,

we can still derive a cut-set lower bound by considering all partitions of the vertices. We also propose an achievable scheme that consists of activating a subtree of edges and using the optimal strategy for transmissions on the tree. While the upper and lower bounds do not match even for very simple functions, for complete graphs, we show that aggregation along trees provides a 2-OPT solution.

A. The two node problem

Consider two nodes 1 and 2 with variables $X_1 \in \{0, 1, \ldots, m_1\}$ and $X_2 \in \{0, 1, \ldots, m_2\}$. Both nodes wish to compute a function $f(X_1, X_2)$ which only depends on the value of $X_1 + X_2$. To put this in context, one can suppose there are m_1 Boolean variables collocated at node 1 and m_2 Boolean variables at node 2, and both nodes wish to compute a symmetric Boolean function of the $n := m_1 + m_2$ variables. We pose the problem in a block computation setting, where each node i has a block of N independent measurements, denoted by X_i^N. We consider the class of all interactive strategies, where nodes 1 and 2 transmit messages alternately with the value of each subsequent message being allowed to depend on all previous transmissions, and the block of measurements available at the transmitting node. We define a round to include one transmission by each node. A strategy is said to achieve correct block computation if for *every* choice of input (X_1^N, X_2^N), each node i can correctly decode the value of the function block $f^N(X_1, X_2)$ using the sequence of transmissions $b_1, b_2, \ldots$ and its own measurement block X_i^N.

Let $\mathscr{S}_N$ be the set of strategies for block length N, which achieve zero-error block computation, and let $C(f, S_N, N)$ be the worst-case total number of bits exchanged under strategy $S_N \in \mathscr{S}_N$. The worst-case per-instance complexity of computing a function $f(X_1, X_2)$ is defined as

$$C(f) := \lim_{N \to \infty} \min_{S_N \in \mathscr{S}_N} \frac{C(f, S_N, N)}{N}.$$

1) Complexity of sum-threshold functions: In this paper, we are only interested in functions $f(X_1, X_2)$ which only depend on $X_1 + X_2$. Let us suppose without loss of generality that $m_1 \le m_2$. We define an interesting class of $\{0, 1\}$-valued functions called sum-threshold functions.

Definition 4 (sum-threshold functions): A sum-threshold function $\Pi_\theta(X_1, X_2)$ with threshold

θ is defined as follows:

$$\Pi_\theta(X_1, X_2) = \begin{cases} 1 & \text{if } X_1 + X_2 \geq \theta, \\ 0 & \text{otherwise.} \end{cases}$$

For the special case where $m_1 = 1, m_2 = 1$ and $\theta = 2$, we recover the Boolean AND function, which was studied in [9]. It is critical to understand this problem before we can address the general problem of computing symmetric Boolean functions. Consider two nodes with measurement blocks $X_1^N \in \{0,1\}^N$ and $X_2^N \in \{0,1\}^N$, which want to compute the element-wise AND of the two blocks, denoted by $\wedge^N(X_1, X_2)$.

Theorem 5: Given any strategy S_N for block computation of $X_1 \wedge X_2$,

$$C(X_1 \wedge X_2, S_N, N) \geq N\log_2 3.$$

Further, there exists a strategy S_N^* which satisfies

$$C(X_1 \wedge X_2, S_N^*, N) \leq \lceil N\log_2 3 \rceil.$$

Thus, the complexity of computing $X_1 \wedge X_2$ is given by $C(X_1 \wedge X_2) = log_2 3$.

Proof of achievability: Suppose node 1 transmits first using a prefix-free codebook. Let the length of the codeword transmitted be $l(X_1^N)$. At the end of this transmission, both nodes know the value of the function at the instances where $X_1 = 0$. Thus node 2 only needs to indicate its bits for the instances of the block where $X_1 = 1$. Thus the total number of bits exchanged under this scheme is $l(X_1^N) + w(X_1^N)$, where $w(X_1^N)$ is the number of 1s in X_1^N. For a given scheme, let us define

$$L := \max_{X_1^N}(l(X_1^N) + w(X_1^N)),$$

to be the worst case total number of bits exchanged. We are interested in finding the codebook which will result in the minimum worst-case number of bits.

Any prefix-free code must satisfy the Kraft inequality given by $\sum_{X_1^N} 2^{-l(X_1^N)} \leq 1$. Consider a codebook with $l(X_1^N) = \lceil N\log_2 3 \rceil - w(x_1^N)$. This satisfies the Kraft inequality since $\sum_{X_1^N} w(X_1^N) = 3^N$. Hence there exists a valid prefix free code for which the worst case number of bits exchanged

is $\lceil N\log_2 3\rceil$, which establishes that $C(X_1 \wedge X_2) \leq \log_2 3$.

The lower bound is shown by constructing a *fooling set* [5] of the appropriate size. We digress briefly to introduce the concept of fooling sets in the context of two-party communication complexity [5]. Consider two nodes X and Y, each of which take values in finite sets $\mathcal{X}$ and $\mathcal{Y}$, and both nodes want to compute some function $f(X,Y)$ with zero error.

Definition 5 (Fooling Set): A set $E \subseteq \mathcal{X} \times \mathcal{Y}$ is said to be a fooling set, if for any two distinct elements $(x_1, y_1), (x_2, y_2)$ in E, we have either

- $f(x_1, y_1) \neq f(x_2, y_2)$, or
- $f(x_1, y_1) = f(x_2, y_2)$, but either $f(x_1, y_2) \neq f(x_1, y_1)$ or $f(x_2, y_1) \neq f(x_1, y_1)$.

Given a fooling set E for a function $f(X_1, X_2)$, we have $C(f(X_1, X_2)) \geq \log_2 |E|$. We have described two dimensional fooling sets above. The extension to multi-dimensional fooling sets is straightforward and gives a lower bound on the communication complexity of the function $f(X_1, X_2, \ldots, X_n)$.

Lower bound for Theorem 5: We define the measurement matrix M to be the matrix obtained by stacking the row X_1^N over the row X_2^N. Thus we need to find a subset of the set of all measurement matrices which forms a fooling set. Let E the set of all measurement matrices which are made up of only the column vectors $\left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$. We claim that E is the appropriate fooling set. Consider two distinct measurement matrices $M_1, M_2 \in E$. Let $f^N(M_1)$ and $f^N(M_2)$ be the block function values obtained from these two matrices. If $f^N(M_1) \neq f^N(M_2)$, we are done. Let us suppose $f^N(M_1) = f^N(M_2)$ and since $M_1 \neq M_2$, there must exist one column where M_1 has $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ but M_2 has $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$. Now if we replace the first row of M_1 with the first row of M_2, the resulting measurement matrix, say M^* is such that $f(M^*) \neq f(M_1)$. Thus, the set E is a valid fooling set. It is easy to verify that the E has cardinality 3^N. Thus, for *any* strategy $S_N \in \mathscr{S}_N$, we must have $C(X_1 \wedge X_2, S_N, N) \geq N\log_2 3$, implying that $C(X_1 \wedge X_2) \geq \log_2 3$. This concludes the proof of Theorem 5. $\square$

We now return to the general two node problem with $X_1 \in \{0, 1, \ldots, m_1\}$ and $X_2 \in \{0, 1, \ldots, m_2\}$ and the sum-threshold function $\Pi_\theta(X_1, X_2)$. We will extend the approach presented above to this

general scenario.

Theorem 6: Given any strategy S_N for block computation of the function $\Pi_\theta(X_1,X_2)$,

$$C(\Pi_\theta(X_1,X_2),S_N,N) \geq N\log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}.$$

Further, there exist single-round strategies S_N^* and S_N^{**}, starting with nodes 1 and 2 respectively, which satisfy

$$C(\Pi_\theta(X_1,X_2),S_N^*,N) \leq \lceil N\log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}\rceil.$$

$$C(\Pi_\theta(X_1,X_2),S_N^{**},N) \leq \lceil N\log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}\rceil.$$

Thus, the complexity of computing $\Pi_\theta(X_1,X_2)$ is given by $C(\Pi_\theta(X_1,X_2)) = \log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}$.

Proof of achievability: We consider three cases:

(a) Suppose $\theta \leq m_1 \leq m_2$. We specify a strategy S_N^* in which node 1 transmits first. We begin by observing that inputs $X_1 = \theta, X_1 = (\theta+1)\ldots,X_1 = m_1$ need not be *separated*, since for each of these values of X_1, $\Pi_\theta(X_1,X_2) = 1$ for all values of X_2. Thus node 1 has an effective alphabet of $\{0,1,\ldots,\theta\}$. Suppose node 1 transmits using a prefix-free codeword of length $l(X_1^N)$. At the end of this transmission, node 2 only needs to indicate one bit for the instances of the block where $X_1 = 0,1,\ldots,(\theta-1)$. Thus the worst-case total number of bits is

$$L := \max_{X_1^N}(l(X_1^N)+w^0(X_1^N)+w^1(X_1^N)+\ldots+w^{\theta-1}(X_1^N)),$$

where $w^j(X_1^N)$ is the number of instances in the block where $X_1 = j$. We are interested in finding the codebook which will result in the minimum worst-case number of bits. From the Kraft inequality for prefix-free codes we have

$$\sum_{X_1^N\in\{0,1,\ldots,\theta\}^N} 2^{-L+w^0(X_1^N)+w^1(X_1^N)+\ldots+w^{\theta-1}(X_1^N))} \leq 1.$$

Consider a codebook with $l(X_1^N) = \lceil N\log_2(2\theta+1)\rceil - w(x_1^N)$. This satisfies the Kraft in-

equality since

$$\sum_{X_1^N \in \{0,1,\ldots,\theta\}^N} 2^{w^0(X_1^N)+w^1(X_1^N)+\ldots+w^{\theta-1}(X_1^N))} .1^{w^\theta(X_1^N)} = (2\theta+1)^N.$$

Hence there exists a prefix-free code for which the worst-case total number of bits exchanged is $\lceil N\log_2(2\theta+1)\rceil$. Since $\theta \le m_1 \le m_2$, we have

$$C(\Pi_\theta(X_1,X_2),S_N^*,N) \le \lceil N\log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}\rceil.$$

The strategy S_N^{**} starting at node 2 can be similarly derived. Node 2 now has an effective alphabet of $\{0,1,\ldots,\theta\}$, and we have $C(\Pi_\theta(X_1,X_2),S_N^{**},N) \le \lceil N\log_2(2\theta+1)\rceil$.

(b) Suppose $m_1 \le m_2 < \theta$. Consider a strategy S_N^* in which node 1 transmits first. The inputs $X_1=0, X_1=1,\ldots,X_1=\theta-m_2-1$ need not be *separated* since for each of these values of X_1, $\Pi_\theta(X_1,X_2)=0$ for all values of X_2. Thus node 1 has an effective alphabet of $\{\theta-m_2-1,\theta-m_2,\ldots,m_1\}$. Upon hearing node 1's transmission, node 2 only needs to indicate one bit for the instances of the block where $X_1=\theta-m_2,\ldots,m_1$. Consider a codebook with $l(X_1^N) = \lceil N\log_2(2(m_1+m_2-\theta+1)+1)\rceil - w^{\theta-m_2}(X_1^N) - \ldots - w^{m_1}(X_1^N)$. This satisfies the Kraft inequality and we have $L = \lceil N\log_2(2(n-\theta+1)+1)\rceil$. Since $m_1 \le m_2 < \theta$, we have that

$$C(\Pi_\theta(X_1,X_2),S_N^*,N) \le \lceil N\log_2\{\min(2\theta+1,2m_1+2,2(n-\theta+1)+1)\}\rceil.$$

The strategy S_N^{**} starting at node 2 can be analogously derived.

(c) Suppose $m_1 < \theta \le m_2$. For the case where node 1 transmits first, we construct a trivial strategy S_N^* where node 1 uses a codeword of length $\lceil N\log_2(m_1+1)\rceil$ bits and node 2 replies with a string of N bits indicating the function block. Thus we have $C(\Pi_\theta(X_1,X_2),S_N^*,N) \le \lceil N\log_2(2m_1+2)\rceil$.

Now consider a strategy S_N^{**} where node 2 transmits first. Observe that the inputs $X_2=0,X_2=1,\ldots,X_2=\theta-m_1-1$ need not be separated since for each of these values of X_2, $\Pi_\theta(X_1,X_2)=0$

for all values of X_2. Further, the inputs $X_2 = \theta, X_2 = \theta + 1, \ldots, X_2 = m_2$ need not be *separated*. Thus node 1 has an effective alphabet of $\{\theta - m_1 - 1, \theta - m_1, \ldots, \theta\}$. Upon hearing node 2's transmission, node 1 only needs to indicate one bit for the instances of the block where $X_2 = \theta - m_1, \ldots, \theta - 1$. Consider a codebook with $l(X_2^N) = \lceil N \log_2(2m_1 + 2) \rceil - w^{\theta - m_1}(X_1^N) - \ldots - w^{\theta - 1}(X_1^N)$. This satisfies the Kraft inequality and we have $L = \lceil N \log_2(2(n - \theta + 1) + 1) \rceil$. Since $m_1 < \theta \le m_2$, we have that

$$C(\Pi_\theta(X_1, X_2), S_N^{**}, N) \le \lceil N \log_2\{\min(2\theta + 1, 2m_1 + 2, 2(n - \theta + 1) + 1)\} \rceil.$$

The lower bound is shown by constructing a *fooling set* as before. Let E denote the set of all measurement matrices which are made up only of the column vectors from the set

$$Z = \left\{ \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} : 0 \le z_1 \le m_1, 0 \le z_2 \le m_2, (\theta - 1) \le z_1 + z_2 \le \theta \right\}.$$

We claim that E is the appropriate fooling set. Consider two distinct measurement matrices $M_1, M_2 \in E$. Let $f^N(M_1)$ and $f^N(M_2)$ be the block function values obtained from these two matrices. If $f^N(M_1) \ne f^N(M_2)$, we are done. Let us suppose $f^N(M_1) = f^N(M_2)$, and note that since $M_1 \ne M_2$, there must exist one column where M_1 and M_2 differ. Suppose M_1 has $\begin{bmatrix} z_{1a} \\ z_{2a} \end{bmatrix}$ while M_2 has $\begin{bmatrix} z_{1b} \\ z_{2b} \end{bmatrix}$, where $z_{1a} + z_{2a} = z_{1b} + z_{2b}$. Assume without loss of generality that $z_{1a} < z_{1b}$ and $z_{2a} > z_{2b}$.

- If $z_{1a} + z_{2a} = z_{1b} + z_{2b} = \theta - 1$, then the *diagonal* element $f(z_{1b}, z_{2a}) = 1$ since $z_{1b} + z_{2a} \ge \theta$. Thus, if we replace the first row of M_1 with the first row of M_2, the resulting measurement matrix, say M^*, is such that $f(M^*) \ne f(M_1)$.

- If $z_{1a} + z_{2a} = z_{1b} + z_{2b} = \theta$, then the *diagonal* element $f(z_{1a}, z_{2b}) = 0$ since $z_{1b} + z_{2a} < \theta$. Thus, if we replace the second row of M_1 with the second row of M_2, the resulting matrix M^* is such that $f(M^*) \ne f(M_1)$.

Thus, the set E is a valid fooling set with cardinality $|Z|^N$. For any strategy S_N, we have $C(f, S_N, N) \geq N \log_2 |Z|$. The cardinality of Z can be modeled as the sum of the coefficients of Y^θ and $Y^{\theta-1}$ in a carefully constructed polynomial:

$$
\begin{aligned}
|Z| &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right](1 + Y + \ldots + Y^{m_1})(1 + Y + \ldots + Y^{m_2}) \\
&= \left[Y^\theta\right] + \left[Y^{\theta-1}\right]\frac{(1 - Y^{m_1+1})(1 - Y^{m_2+1})}{(1-Y)^2}.
\end{aligned}
$$

This is solved using the binomial expansion for $\frac{1}{(1-Y)^k}$ [21].

$$
|Z| = \left[Y^\theta\right] + \left[Y^{\theta-1}\right](1 - Y^{m_1+1})(1 - Y^{m_2+1}) \sum_{k=0}^{\infty} \binom{k+1}{1} Y^k.
$$

(a) Suppose $\theta \leq m_1 \leq m_2$. Then $|Z| = \theta + \theta + 1$.

(b) Suppose $m_1 \leq \theta \leq m_2$. Then $|Z| = 2m_1 + 2$.

(c) Suppose $m_1 \leq m_2 \leq \theta$. Then $|Z| = 2(n - \theta + 1) + 1$.

This completes the proof of Theorem 6. □

2) Complexity of sum-interval functions:

Definition 6 (sum-interval functions): A sum-interval function $\Pi_{[a,b]}(X_1, X_2)$ on the interval $[a, b]$ is defined as follows:

$$
\Pi_{[a,b]}(X_1, X_2) := \begin{cases} 1 & \text{if } a \leq X_1 + X_2 \leq b, \\ 0 & \text{otherwise.} \end{cases}
$$

Theorem 7: Given any strategy S_N for block computation of $\Pi_{[a,b]}(X_1, X_2)$ where $b \leq n/2$,

$$
C(\Pi_{[a,b]}(X_1, X_2), S_N, N) \geq N \log_2\{\min(2b - a + 3, m_1 + 1)\}.
$$

Further, there exists a single-round strategy S_N^* which satisfies

$$
C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2\{\min(2(b+1) + 1, 2m_1 + 2)\}\rceil.
$$

Thus, we have obtained the complexity of computing $\Pi_\theta(X_1, X_2)$ to within one bit.

Proof of Achievability:

(a) Suppose $b \leq m_1 \leq m_2$. Node 1 has an effective alphabet of $\{0, 1, \ldots, b+1\}$. Then the worst-case total number of bits exchanged is given by

$$L := \max_{X_1^N}(l(X_1^N) + w^0(X_1^N) + \ldots + w^b(X_1^N)).$$

From the Kraft inequality, we can obtain a prefix free codebook with $L = \lceil N \log_2(2b+1) + 1) \rceil$. Thus we have

$$C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2(2(b+1) + 1) \rceil.$$

(b) Suppose $m_1 \leq a \leq b \leq m_2$ or $a \leq m_1 \leq b \leq m_2$. In either of these scenarios, node 1 has an effective alphabet of $\{0, 1, \ldots, m_1\}$. Then the worst-case total number of bits exchanged is given by

$$L := \max_{X_1^N}(l(X_1^N) + w^{a-m_2}(X_1^N) + \ldots + w^{m_1}(X_1^N))$$

From the Kraft inequality, we can obtain a prefix free codebook with $L = \lceil N \log_2(2m_1 + 2) \rceil$. Thus we have $C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2(2m_1 + 2) \rceil$.

Proof of Lower Bound: We attempt to find a fooling subset E of the set of measurement matrices. Our first guess would be the set of measurement matrices which are composed of only column vectors which sum up to b or $b+1$. However we see that this is not necessarily a fooling set, because if $[z_{1a}, z_{2a}]^T$ and $[z_{1b}, z_{2b}]^T$ are two columns which sum to $b+1$, and if $z_{1a} \leq z_{1b} - (b-a+2)$, then neither of the diagonal elements evaluate to function value 1. Thus, we can pick a maximum of $(b-a+2)$ consecutive elements along the line $z_1 + z_2 = b+1$, and, as before, all the elements on the line $z_1 + z_2 = b$. It is easy to check that this modified set of columns indeed yields a fooling set of measurement matrices. Now we need to compute the number of such columns.

(a) Suppose $b \leq m_1 \leq m_2$. The number of columns which sum up to b is equal to $b+1$. Thus the size of the fooling set is given by $(2b - a + 3)^N$.

(b) Suppose $a \leq m_1 \leq b \leq m_2$ or $m_1 \leq a \leq b \leq m_2$. The number of columns which sum up to

b is equal to $m_1 + 1$ and the number of columns which sum up to $b+1$ is equal to $m_1 + 1$. Thus, the size of the fooling set is given by $\{(m_1 + 1) + \min(m_1 + 1, b - a + 2)\}^N$.

3) A general strategy for achievability: The strategy for achievability used in Theorems 6 and 7 suggests an achievable scheme for any general function $f(X_1, X_2)$ of variables $X_1 \in \mathcal{X}_1$ and $X_2 \in \mathcal{X}_2$ which depends only on the value of $X_1 + X_2$. This is done in two stages.

- **Separation:** Two inputs x_{1a} and x_{1b} need not be *separated* if $f(x_{1a}, x_2) = f(x_{1b}, x_2)$ for all values x_2. By checking this condition for each pair (x_{1a}, x_{1b}), we can arrive at a partition of $\{0, 1 \ldots, m_1\}$ into equivalence classes, which can be considered a reduced alphabet, say $A := \{a_1, \ldots, a_l\}$.

- **Coding:** Let A_0 denote the subset of the alphabet A for which the function evaluates only to 0, irrespective of the value of X_2, and let A_1 denote the subset of A which always evaluates to 1. Clearly, from the equivalence class structure, we have $|A_0| \leq 1$ and $|A_1| \leq 1$. Using the Kraft inequality as in Theorems 6 and 7, we obtain a scheme S_N^* with complexity $\log_2(2l - |A_0| - |A_1|)$.

B. Computing symmetric Boolean functions on tree networks

Consider a tree graph $T = (V, E)$, with node set $V = \{0, 1, \ldots, n\}$ and edge set E. Each node i has a Boolean variable $X_i \in \{0, 1\}$, and every node wants to compute a given symmetric Boolean function $f(X_1, X_2, \ldots, X_n)$. Again, we allow for block computation and consider all strategies where nodes can transmit in any sequence with possible repetitions, subject to:

- On any edge $e = (i, j)$, either node i transmits or node j transmits, or neither, and this is determined from the previous transmissions.

- Node i's transmission can depend on the previous transmissions and the measurement block X_i^N.

For sum-threshold functions, we have a computation and communication strategy that is optimal for each link.

Theorem 8: Consider a tree network where we want to compute the function $\Pi_\theta(X_1, \ldots, X_n)$. Let us focus on a single edge $e \equiv (i, j)$ whose removal disconnects the graph into components

A_e and $V \setminus A_e$, with $|A_e| \leq |V \setminus A_e|$. For any strategy $S_N \in \mathscr{S}_N$, the number of bits exchanged along edge $e \equiv (i,j)$, denoted by $C_e(\Pi_\theta(X_1,\ldots,X_n),S_N,N)$, is lower bounded by

$$C_e(\Pi_\theta(X_1,\ldots,X_n),S_N,N) \geq N\log_2\{\min(2\theta+1,2|A_e|+2,2(n-\theta+1)+1)\}.$$

Further, there exists a strategy S_N^* such that for any edge e,

$$C_e(\Pi_\theta(X_1,\ldots,X_n),S_N^*,N) \leq \lceil N\log_2\{\min(2\theta+1,2|A_e|+2,2(n-\theta+1)+1)\} \rceil.$$

The complexity of computing $\Pi_\theta(X_1,\ldots,X_n)$ is given by

$$C_e(\Pi_\theta(X_1,\ldots,X_n)) = \log_2\{\min(2\theta+1,2|A_e|+2,2(n-\theta+1)+1)\}.$$

Proof: Given a tree network T, every edge e is a cut edge. Consider an edge e whose removal creates components A_e and $V \setminus A_e$, with $|A_e| \leq |V \setminus A_e|$. Now let us aggregate the nodes in A_e and also those in $V \setminus A_e$, and view this as a problem with two nodes connected by edge e. Clearly the complexity of computing the function $\Pi_\theta(X_{A_e},X_{V \setminus A_e})$ is a lower bound on the worst-case total number of bits that must be exchanged on edge e under any strategy S_N. Hence we obtain

$$C_e(\Pi_\theta(X_1,\ldots,X_n),S_N,N) \geq N\log_2\{\min(2\theta+1,2|A_e|+2,2(n-\theta+1)+1)\}.$$

The achievable strategy S_N^* is derived from the achievable strategy for the two node case in Theorem 6. While the transmissions back and forth along any edge will be exactly the same, we need to orchestrate these transmissions so that conditions of causality are maintained. Pick any node, say r, to be the root. This induces a partial order on the tree network. We start with each leaf in the network transmitting its codeword to the parent. Once a parent node obtains a codeword from each of its children, it has sufficient knowledge to disambiguate the letters of the effective alphabet of the subtree, and subsequently it transmits a codeword to its parent. Thus codewords are transmitted from child nodes to parent nodes until the root is reached. The root can then compute the value of the function and now sends the appropriate replies to its children. The children then compute the function and send appropriate replies, and so on. This

sequential strategy depends critically on the fact that, in the two node problem, we derived optimal strategies starting from either node. For any edge e, the worst-case total number of bits exchanged is given by

$$C_e(\Pi_\theta(X_1,\ldots,X_n),S_N^*,N) \le \lceil N\log_2\{\min(2\theta+1,2|A_e|+2,2(n-\theta+1)+1)\}\rceil. \square$$

One can similarly derive an approximately optimal strategy for sum-interval functions, which we state here without proof.

Theorem 9: Consider a tree network where we want to compute the function $\Pi_{[a,b]}(X_1,\ldots,X_n)$, with $b \le \frac{n}{2}$. Let us focus on a single edge $e \equiv (i,j)$ whose removal disconnects the graph into components A_e and $V \setminus A_e$, with $|A_e| \le |V \setminus A_e|$. For any strategy $S_N \in \mathscr{S}_N$, the number of bits exchanged along edge $e \equiv (i,j)$, denoted by $C_e(f,S_N,N)$ is lower bounded by

$$C_e(\Pi_{[a,b]}(X_1,\ldots,X_n),S_N,N) \ge N\log_2\{\min(2b-a+3,|A_e|+1)\}.$$

Further there exists a strategy S_N^* such that for any edge e,

$$C_e(\Pi_{[a,b]}(X_1,\ldots,X_n),S_N^*,N) \le \lceil N\log_2\{\min(2(b+1)+1,2|A_e|+2)\}\rceil.$$

C. Extension to non-binary alphabets

The extension to the case where each node draws measurements from a non-binary alphabet is immediate. Consider a tree network with n nodes where node i has a measurement $X_i \in \{0,1,\ldots,l_i-1\}$. Suppose all nodes want to compute a given function which only depends on the value of $X_1+X_2+\ldots+X_n$. We can define sum-threshold functions in analogous fashion and derive an optimal strategy for computation.

Theorem 10: Consider a tree network where we want to compute a sum-threshold function, $\Pi_\theta(X_1,\ldots,X_n)$, of non-binary measurements. Let us focus on a single edge e whose removal disconnects the graph into components A_e and $V \setminus A_e$. Let us define $l_{A_e} := \sum_{i\in A_e} l_i$. Then the complexity of computing $\Pi_\theta(X_1,\ldots,X_n)$ is given by

$$C_e(\Pi_\theta(X_1,\ldots,X_n)) = \log_2\{\min(2\theta+1,2\min(l_{A_e},l_{V\setminus A_e})+2,2(l_V-\theta+1)+1)\}.$$

Theorem 9 also extends to the case of non-binary alphabets.

D. Computing sum-threshold functions in general graphs

We now consider the computation of sum-threshold functions in general graphs where the alphabet is not restricted to be binary. A cut is defined to be a set of edges $F \subseteq E$ which disconnect the network into two components A_F and $V \setminus A_F$.

Lemma 5 (Cut-set bound): Consider a general network $G = (V, E)$, where node i has measurement $X_i \in \{0, 1, \ldots, l_i - 1\}$ and all nodes want to compute the function $\Pi_\theta(X_1, \ldots, X_n)$. Given a cut F which separates A_F from $V \setminus A_F$, the cut-set lower bound specifies that: For any strategy S_N, the number of bits exchanged on the edges in F is lower bounded by

$$C_F(\Pi_\theta(X_1, \ldots, X_n), S_N, N) \geq N \log_2(\min\{2\theta + 1, 2m_F + 2, 2(l_V - \theta + 1) + 1)\}.$$

where $l_{A_F} = \sum_{i \in A_F} l_i$ and $m_F = \min(l_{A_F}, l_{V \setminus A_F})$.

A natural achievable strategy is to pick a spanning subtree of edges and use the optimal strategy on this subtree. The convex hull of the rate vectors of the subtree aggregation schemes, is an achievable region. We wish to compare this with the cut-set region. To simplify matters, consider a complete graph G where each node i has a measurement $X_i \in \{0, \ldots, l - 1\}$. Let R_{ach} be the maximum symmetric ratepoint achievable by aggregating along trees, and R_{cut} be the minimum symmetric ratepoint that satisfies the cut-set constraints.

Theorem 11: For the computation of sum-threshold functions on complete graphs, $R_{ach} \leq 2(1 - \frac{1}{n})) R_{cut}$. In fact, this approximation ratio is tight.

Proof: Let us assume without loss of generality that $\theta \leq \frac{n.l}{2}$. Consider all cuts of the type $(\{i\}, V \setminus \{i\})$. This yields

$$R_{cut} \geq \max_{i \in V} \left(\frac{\min(\log_2(2\theta + 1), \log_2(2l_i + 2))}{n - 1} \right).$$

Now consider the achievable scheme which employs each of the n star graphs for equal sized sub-blocks of measurements. The rate on edge (i,j) is given by

$$\frac{1}{n}\left(\min(\log_2(2\theta+1),\log_2(2l_i+2))+\min(\log_2(2\theta+1),\log_2(2l_j+2))\right)$$

Hence we have

$$R_{ach} \le \frac{2}{n}(\min(\log_2(2\theta+1),\max_{i\in V}\{\log_2(2l_i+2)\})) \le 2\left(1-\frac{1}{n}\right)R_{cut}.$$

Tight Example: Suppose $l_1 = l_2 = \ldots = l_n = l$ and $\theta > l$, then

$$R_{cut} = \frac{1}{n-1}\min(\log_2(2\theta+1),\log_2(2l+2))$$

Further, from the symmetry of the problem, it is clear that the optimal scheme is to employ the n star graphs for equal sub-blocks of measurements. This gives a symmetric achievable point of

$$R_{ach} = \frac{2}{n}\min(\log_2(2\theta+1),\log_2(2l+2)) = 2\left(1-\frac{1}{n}\right)R_{cut}.$$

E. Linear Programming Formulation

The above approach of restricting attention to aggregation along star graphs, gives in to a convenient Linear Programming (LP) formulation. Consider a complete graph G. Let us define the rate region achievable by star graphs in the following way

$$\tilde{\mathscr{R}}_{ach} = \{A\underline{\lambda} : ||\lambda||_1 = 1\}$$

where A is a $n \times \frac{n(n-1)}{2}$ matrix where a_{ie}th entry is the minimum number of bits that must be sent along edge e under tree aggregation scheme T_i. The vector $\underline{\lambda}$ is the relative weights assigned to the different trees. We want to compare the rate vectors achieved by this scheme with the rate vectors that satisfy the cut constraints. Let $\underline{r} \in \mathscr{R}_{cut}$ be a given rate vector which satisfies the cut constraints of Lemma 1. Now, we seek to find an achievable rate vector that is within a θ factor of $\underline{r}$, and further, we want to find the minimum value of such a θ. This can formulated as a linear program

$$\text{Min. } \theta$$
$$\text{s.t. } A\underline{\lambda} \leq \theta\underline{r}$$
$$||\lambda||_1 \geq 1$$
$$\lambda \geq 0, \, \theta \geq 0$$

Thus we can obtain the optimal assignment λ^* and the optimal factor θ^*. Note that this assignment depends on the given rate vector $\underline{r} \in \mathscr{R}_{cut}$. We can also write similar such LPs for other classes of trees.

V. CONCLUDING REMARKS

In this paper, we have addressed some problems that arise in the context of information aggregation in sensor networks. While the general problem of devising optimal strategies for function computation in wireless networks appears formidable, we have simplified it by abstracting out the medium access control problem and analyzing the problem of function computation in graphs.

We have started with the problem of zero error function computation in directed graphs, and analyzed both worst case and average case metrics. For directed tree graphs, we have constructed optimal encoding schemes on each edge. This matches the cut-set lower bounds. For general DAGs, we have provided an outer bound on the rate region, and an achievable region based on aggregating along subtrees. While we have presented some examples where tree aggregation schemes are optimal, it remains to quantify the sub-optimality of tree aggregation schemes in general.

We have also addressed the computation of symmetric Boolean functions in undirected graphs, where all nodes want to compute the function. For the case of computing sum-threshold functions in undirected trees, we have derived the optimal strategy for each edge. The achievable scheme for block computation involves a layering of transmissions that is reminiscent of message passing. Our framework can be generalized to handle functions of integer measurements which only depend on the sum of the measurements. The extension to general graphs is very interesting and appears significantly harder. However, a cut-set lower bound can be immediately derived, and in some special cases one can show that subtree aggregation schemes provide a 2-OPT solution. Once again, it remains to study the suboptimality of tree aggregation schemes in general graphs.

REFERENCES

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, July 2000.

[2] R. Appuswamy, M. Franceschetti, N. Karamchandani, and K. Zeger. Network coding for computing. In *Proceedings of the 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 1–6, September 2008.

[3] A. Giridhar and P. R. Kumar. Computing and communicating functions over sensor networks. *IEEE Journal on Selected Areas in Communication*, 23(4):755–764, April 2005.

[4] S. Subramanian, P. Gupta, and S. Shakkottai. Scaling bounds for function computation over large networks. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pages 136–140, June 2007.

[5] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997.

[6] I. Wegener. *The Complexity of Boolean Functions*. J. Wiley & Sons, Inc., New York, NY, USA, 1987.

[7] A. Orlitsky and A. El Gamal. Average and randomized communication complexity. *IEEE Transactions on Information Theory*, 36:3–16, 1990.

[8] M. Karchmer, R. Raz, and A. Wigderson. Super-logarithmic depth lower bounds via direct sum in communication coplexity. In *Structure in Complexity Theory Conference*, pages 299–304, 1991.

[9] R. Ahlswede and Ning Cai. On communication complexity of vector-valued functions. *IEEE Transactions on Information Theory*, 40:2062–2067, 1994.

[10] F. R. Kschischang, B. J. Frey, and H. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[11] S. Aji and R. Mceliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, 2000.

[12] A. D. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1):1–10, January 1976.

[13] A. Orlitsky and J. R. Roche. Coding for computing. *IEEE Transactions on Information Theory*, 47:903–917, 2001.

[14] H. Witsenhausen. The zero-error side information problem and chromatic numbers. *IEEE Transactions on Information Theory*, 22:592–593, September 1976.

[15] N. Alon and A. Orlitsky. Source coding and graph entropies. *IEEE Transactions on Information Theory*, 42:1329–1339, September 1996.

[16] N. Ma and P. Ishwar. Two-terminal distributed source coding with alternating messages for function computation. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pages 51–55, 2008.

[17] N. Ma, P. Ishwar, and P. Gupta. Information-theoretic bounds for multiround function computation in collocated networks. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pages 2306–2310, 2009.

[18] R. D. Gallager. Finding parity in a simple broadcast network. *IEEE Transactions on Information Theory*, 34(2):176–180, March 2008.

[19] L. Ying, R. Srikant, and G. E. Dullerud. Distributed symmetric function computation in noisy wireless sensor networks. *IEEE Transactions on Information Theory*, 53(12):4826–4833, December 2007.

[20] C. Dutta, Y. Kanoria, D. Manjunath, and J. Radhakrishnan. A tight lower bound for parity in noisy communication networks. In *Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1056–1065, January 2008.

[21] D. West. *Combinatorial Mathematics*. Course notes for ECE 580, Department of Mathematics, University of Illinois at Urbana-Champaign, 2008.